Convolutional Networks for Images, Speech, and Time-Series

Y. Bengio
Université de Montréal
889 PUBLICATIONS 457,579 CITATIONS

Yann Lecun
New York University
571 PUBLICATIONS 177,301 CITATIONS

Convolutional Networks for Images, Speech, and Time-Series

Yann LeCun
Rm 4G332, AT&T Bell Laboratories
101 Crawfords Corner Road
Holmdel, NJ 07733
yann@research.att.com

Yoshua Bengio
Dept. Informatique et Recherche
Opérationnelle, Université de Montréal,
Montreal, Qc, Canada, H3C-3J7
bengioy@iro.umontreal.ca

RUNNING HEAD: Convolutional Networks

Correspondance:

Yann LeCun

Rm 4G332, AT&T Bell Laboratories, 101 Crawfords Corner Road

Holmdel, NJ 07733 , phone: 908-949-4038, fax: 908-949-7322

email: yann@research.att.com

1 INTRODUCTION

The ability of multilayer back-propagation networks to learn complex, high-dimensional, non-linear mappings from large collections of examples makes them obvious candidates for image recognition or speech recognition tasks (see PATTERN RECOGNITION AND NEURAL NETWORKS). In the traditional model of pattern recognition, a hand-designed feature extractor gathers relevant information from the input and eliminates irrelevant variabilities. A trainable classifier then categorizes the resulting feature vectors (or strings of symbols) into classes. In this scheme, standard, fully-connected multilayer networks can be used as classifiers. A potentially more interesting scheme is to eliminate the feature extractor, feeding the network with "raw" inputs (e.g. normalized images), and to rely on backpropagation to turn the first few layers into an appropriate feature extractor. While this can be done with an ordinary fully connected feed-forward network with some success for tasks such as character recognition, there are problems.

Firstly, typical images, or spectral representations of spoken words, are large, often with several hundred variables. A fully-connected first layer with, say a few 100 hidden units, would already contain several 10,000 weights. Overfitting problems may occur if training data is scarce. In addition, the memory requirement for that many weights may rule out certain hardware implementations. But, the main deficiency of unstructured nets for image or speech aplications is that they have no built-in invariance with respect to translations, or local distortions of the inputs. Before being sent to the fixed-size input layer of a neural net, character images, spoken word spectra, or other 2D or 1D signals, must be approximately size-normalized and centered in the input field. Unfortunately, no such preprocessing can be perfect: handwriting is often normalized at the word level, which can cause size, slant, and position variations for individual characters; words can be spoken at varying speed, pitch, and intonation. This will cause variations in the position of distinctive features in input objects. In principle, a fully-connected network of sufficient size could learn to produce outputs that are invariant with respect to such variations. However, learning such a task would probably result in multiple units with identical weight patterns positioned at various locations in the input. Learning these weight configurations requires a very large number of training instances to cover the space of possible variations. On the other hand, in convolutional networks, shift invariance is automatically obtained by forcing the replication of weight configurations across space.

Secondly, a deficiency of fully-connected architectures is that the topology of the input is entirely ignored. The input variables can be presented in any (fixed) order without affecting the outcome of the training. On the contrary, images, or spectral representations of speech have a strong 2D local structure, time-series have a strong 1D structure: variables (or pixels) that are spatially or temporally nearby are highly correlated. Local correlations are the reasons for the well-known advantages of extracting and combining *local* features before recognizing spatial or temporal objects. Convolutional networks force the extraction of local features by restricting the receptive fields of hidden units to be local.

2 CONVOLUTIONAL NETWORKS

Convolutional networks combine three architectural ideas to ensure some degree of shift and distortion invariance: local receptive fields, shared weights (or weight replication), and, sometimes, spatial or temporal subsampling. A typical convolutional network for recognizing characters is shown in figure 1 (from (LeCun et al., 1990)). The input plane receives images of characters that are approximately size-normalized and centered. Each unit of a layer receives inputs from a set of units located in a small neighborhood in the previous layer. The idea of connecting units to local receptive fields on the input goes back to the Perceptron in the early 60s, and was almost simultaneous with Hubel and Wiesel's discovery of locally-sensitive, orientation-selective neurons in the cat's visual system. Local connections have been reused many times in neural models of visual learning (see (Mozer, 1991; Le Cun, 1986) and NEOCOGNITRON in this handbook). With local receptive fields, neurons can extract elementary visual features such as oriented edges, end-points, corners (or similar features in speech spectrograms). These features are then combined by the higher layers. As stated earlier, distortions or shifts of the input can cause the position of salient features to vary. In addition, elementary feature detectors that are useful on one part of the image are likely to be useful across the entire image. This knowledge can be applied by forcing a set of units, whose receptive fields are located at different places on the image, to have identical weight vectors (Rumelhart, Hinton and Williams, 1986). The outputs of such a set of neurons constitute a *feature map*. At each position, different types of units in different feature maps compute different types of features. A sequential implementation of this, for each feature map, would be to scan the input image with a single neuron that has a local receptive field, and to store the states of this neuron at corresponding locations in the feature map. This operation is equivalent to a convolution with a small size kernel, followed by a squashing function. The process can be performed in parallel by implementing the feature map as a plane of neurons that *share* a single weight vector. Units in a feature map are constrained to perform the same operation on different parts of the image. A convolutional layer is usually composed of several feature maps (with different weight vectors), so that multiple features can be extracted at each location. The first hidden layer in figure 1 has 4 feature maps with 5 by 5 receptive fields. Shifting the input of a convolutional layer will shift the output, but will leave it unchanged otherwise. Once a feature has been detected, its exact location becomes less important, as long as its approximate position relative to other features is preserved. Therefore, each convolutional layer is followed by an additional layer which performs a local averaging and a subsampling, reducing the resolution of the feature map, and reducing the sensitivity of the output to shifts and distortions. The second hidden layer in figure 1 performs 2 by 2 averaging and subsampling, followed by a trainable coefficient, a trainable bias, and a sigmoid. The trainable coefficient and bias control the effect of the squashing non-linearity (for example, if the coefficient is small, then the neuron operates in a quasi-linear mode). Successive layers of convolutions and subsampling are typically alternated, resulting in a "bi-pyramid": at each layer, the number of feature maps is increased as the spatial resolution is decreased. Each unit in the third hidden layer in figure 1 may have input connections from several feature maps in the previous layer. The convolution/subsampling combination, inspired by Hubel and Wiesel's notions of "simple" and "complex" cells, was implemented in the Neocognitron model (see NEOCOGNITRON), though no globally supervised learning procedure such as back-propagation was available then.

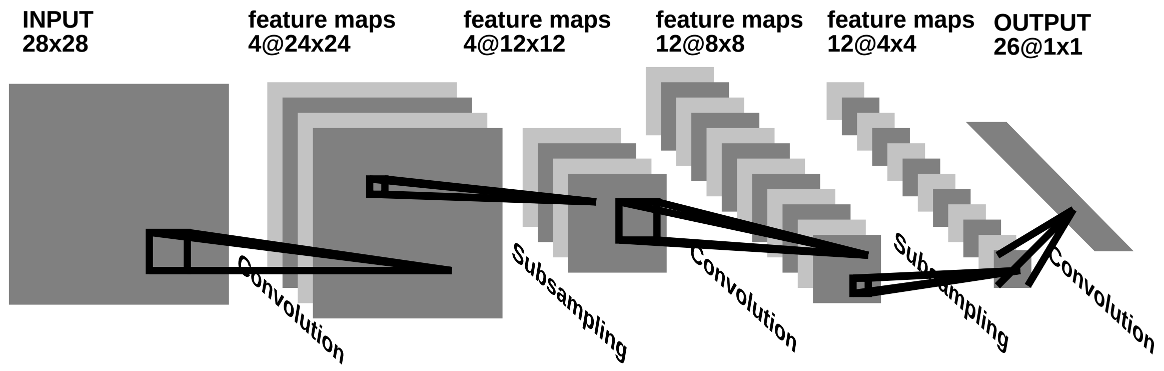

Figure 1: Convolutional Neural Network for image processing, e.g., handwriting recognition

Since all the weights are learned with back-propagation, convolutional networks can be seen as synthesizing their own feature extractor. The weight sharing technique has the interesting side effect of reducing the number of free parameters, thereby reducing the "capacity" of the machine and improving its generalization ability (see (LeCun, 1989) on weight sharing, and LEARNING AND GENERALIZATION for an explanation of notions of capacity and generalization). The network in figure 1 contains about 100,000 connections, but only about 2,600 free parameters because of the weight sharing. Such networks compare favorably with other methods on handwritten character recognition tasks (Bottou et al., 1994) (see also HAND WRITTEN DIGIT RECOGNITION), and they have been deployed in commercial applications.

Fixed-size convolutional networks that share weights along a single temporal dimension are known as Time-Delay Neural Networks (TDNNs). TDNNs have been used in phoneme recognition (without subsampling) (Lang and Hinton, 1988; Waibel et al., 1989), spoken word recognition (with subsampling) (Bottou et al., 1990), and on-line handwriting recognition (Guyon et al., 1991).

3 VARIABLE-SIZE CONVOLUTIONAL NETWORKS, SDNN

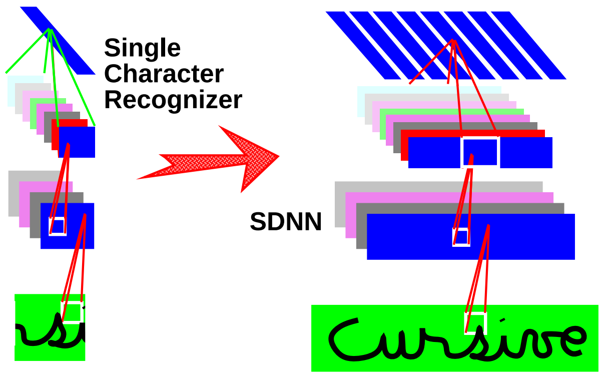

Figure 2: Variable-size replicated convolutional network, SDNN

While characters or short spoken words can be size-normalized and fed to a fixed-size network, more complex objects such as written or spoken words and sentences have inherently variable size. One way of handling such a composite object is to segment it heuristically into simpler objects that can be recognized individually (e.g., characters, phonemes). However, reliable segmentation heuristics do not exist for speech or cursive handwriting. A brute force solution is to scan (or replicate) a recognizer at all possible locations accross the input. While this can be prohibitively expensive in general, convolutional networks can be scanned or replicated very efficiently over large, variable-size input fields. Consider one instance of a convolutional net and its *alter ego* at a nearby location. Because of the convolutional nature of the networks, units in the two nets that look at identical locations on the input have identical outputs, therefore their output does not need to be computed twice. In effect, replicating a convolution network can simply be done by increasing the size of the field over which the convolutions are performed, and replicating the output layer, effectively making it a convolutional layer. An output whose receptive field is centered on an elementary object will produce the class of this object, while an in-between output may be empty or contain garbage. The outputs can be interpreted as evidence for the categories of object centered at different positions of the input field. A post-processor is therefore required to pull out consistant interpretations of the output. Hidden Markov Models (HMM) or other graph-based methods are often used for that purpose (see SPEECH RECOGNITION, and PATTERN RECOGNITION AND NEURAL NETWORKS in this volume). The replicated network and the HMM can be trained simultaneously by back-propagating gradients through the HMM. Globally trained, variable-size TDNN/HMM hybrids have been used for speech recognition (see PATTERN RECOGNITION AND NEURAL NETWORKS for a list of references) and on-line handwriting recognition (Schenkel et al., 1993). Two-dimensional replicated convolutional networks, called "Space Displacement Neural Networks" (SDNN) have been used in combination with HMM or other elastic matching methods for handwritten word recognition (Keeler and Rumelhart, 1991; Matan et al., 1992; Bengio, LeCun and

Henderson, 1994). Another interesting application of SDNNs is object spotting (Wolf and Platt, 1994).

An important advantage of convolutional neural networks is the ease with which they can be implemented in hardware. Specialized analog/digital chips have been designed and used in character recognition, and in image preprocessing applications (Boser et al., 1991). Speeds of more than 1000 characters per second were obtained with a network with around 100,000 connections (shown in figure 1).

The idea of subsampling can be turned around to construct networks similar to TDNNs, but that can generate sequences from labels. These networks are called reverse-TDNNs because they can be viewed as upside-down TDNNs: temporal resolution increases from the input to the output, through alternated oversampling and convolution layers (Simard and LeCun, 1992).

4 DISCUSSION

Convolutional neural networks are a good example of an idea inspired by biology that resulted in competitive engineering solutions that compare favorably with other methods (Bottou et al., 1994). While applying convolutional nets to image recognition removes the need for a separate hand-crafted feature extractor, normalizing the images for size and orientation (if only approximately) is still required. Shared weights and subsampling bring invariance with respect to small geometric transformations or distortions, but fully invariant recognition is still beyond reach. Radically new architectural ideas, possibly suggested by biology, will be required for a fully neural image or speech recognition system.

Acknowledgements

The authors wish to thank Leon Bottou, Chris Burges, Isabelle Guyon, Larry Jackel, and the other members of the Adaptive Systems Research Department for their support and comments.

References

Bengio, Y., LeCun, Y., and Henderson, D. (1994). Globally Trained Handwritten Word Recognizer using Spatial Representation, Space Displacement Neural Networks and Hidden Markov Models. In *Advances in Neural Information Processing Systems*, volume 6, pages 937–944.

Boser, B., Sackinger, E., Bromley, J., LeCun, Y., and Jackel, L. (1991). An analog neural network processor with programmable topology. *IEEE Journal of Solid-State Circuits*, 26(12):2017–2025.

Bottou, L., Cortes, C., Denker, J., Drucker, H., Guyon, I., Jackel, L., LeCun, Y., Muller, U., Sackinger, E., Simard, P., and Vapnik, V. (1994). Comparison of classifier methods: a case study in handwritten digit recognition. In *International Conference on Pattern Recognition*, Jerusalem, Israel.

Bottou, L., Fogelman-Soulie, F., Blanchet, P., and Lienard, J. S. (1990). Speaker independent isolated digit recognition: multilayer perceptrons vs dynamic time warping. *Neural Networks*, 3:453–465.

Guyon, I., Albrecht, P., Le Cun, Y., Denker, J. S., and ubbard W., H. (1991). design of a neural network character recognizer for a touch termin al. *Pattern Recognition*, 24(2):105–119.

Keeler, J. and Rumelhart, D.and Leow, W. (1991). integrated segmentation and recognition of hand-printed numerals. In Lippman, R. P., Moody, J. M., and Touretzky, D. S., editors, *Neural Information Processing Systems*, volume 3, pages 557–563. Morgan Kaufmann Publishers, San Mateo, CA.

Lang, K. and Hinton, G. (1988). The development of the Time-Delay Neural Network architecture for speech recognition. Technical Report CMU-CS-88-152, Carnegie-Mellon University.

Le Cun, Y. (1986). Learning Processes in an Asymmetric Threshold Network. In Bienenstock, E., Fogelman-Soulié, F., and Weisbuch, G., editors, *Disordered systems and biological organization*, pages 233–240, Les Houches, France. Springer-Verlag.

LeCun, Y. (1989). Generalization and Network Design Strategies. Technical Report CRG-TR-89-4, Department of Computer Science, University of Toronto.

LeCun, Y., Boser, B., Denker, J., Henderson, D., Howard, R., Hubbard, W., and Jackel, L. (1990). Handwritten Digit Recognition with a Back-Propagation Network. In Touretzky, D., editor, *Advances in Neural Information Processing Systems*, volume 2, pages 396–404, Denver 1989. Morgan Kaufmann, San Mateo.

Matan, O., Burges, C., LeCun, Y., and Denker, J. (1992). Multi-Digit Recognition Using a Space Displacement Neural Network. In Moody, J., Hanson, S., and Lipmann, R., editors, *Advances in Neural Information Processing Systems 4*, pages 488–495, San Mateo CA. Morgan Kaufmann.

Mozer, M. (1991). *The Perception of Multiple Objects, A Connectionist Approach.* MIT Press.

Rumelhart, D., Hinton, G., and Williams, R. (1986). Learning Representations by Back-Propagating Errors. *Nature*, 323:533–536.

Schenkel, M., Weissman, H., Guyon, I., Nohl, C., and Henderson, D. (1993). Recognition-Based Segmentation of On-Line Hand-Printed Words. In Hanson, C. and Giles, editors, *Advances in Neural Information Processing Systems*, volume 5, pages 723–730, Denver, CO.

Simard, P. and LeCun, Y. (1992). Reverse TDNN: An Architecture for Trajectory Generation. In Moody, J., Hanson, S., and Lipmann, R., editors, *Advances in Neural Information Processing Systems 4*, pages 579–588, Denver 1991. Morgan Kaufmann, San Mateo.

Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., and Lang, K. (1989). Phoneme Recognition Using Time-Delay Neural Networks. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37:328–339.

Wolf, R. and Platt, J. (1994). Postal address block location using a convolutional locator network. In *Advances in Neural Information Processing Systems 6*, pages 745–752.